Pergamon

Computers Math. Applic. Vol. 28, No. 5, pp. 75–88, 1994

0898-1221/94 $7.00 + 0.00

0898-1221(94)00146-4

Computer Simulations of Path Generation and Path Form Modification with Local Rules Working on a Parallel Cell-Based Architecture

K. DAUTENHAHN AND H. CRUSE*
Department of Biological Cybernetics, Faculty of Biology
University of Bielefeld, Postfach 100131, D-33501 Bielefeld, Germany

(Received August 1993; accepted September 1993)

Abstract—Many mathematical and engineering algorithms are dealing with path finding and path planning problems which arise for a mobile navigating robot or the endeffector of a manipulator. Natural mobile systems, like animals navigating in their everyday surroundings, are faced with the same problems and normally generate adequate solutions. For this reason we performed psychometric experiments with human probands [1,2] to investigate both decision-making and path generation in the case of full information. In this paper we present several methods which can be used to generate path forms roughly comparable to those produced by the human probands. The computer simulations proposed here concentrate on algorithms which use local rules and work upon a parallel, cell-based architecture. On the basis of the 'dynamic-system metaphor,' we define a framework for the investigation of several algorithms. A diffusion-algorithm proposed in [3,4] is modified and combined with a so-called relative potential field. This diffusion in potential fields (DIP) provides a method to influence the smoothness of the forms of the path during the development of the gradient field. Once a path has been found, the relative potentials can also help to gradually modulate the first found path with a time-saving mechanism called Tube-Diffusion. In addition to the DIP-algorithm, we investigated two wave-spreading algorithms which generate cost-optimal paths. These are a further development of the path planning approach used in [5] and a new approach which we called Inclusive-Or-algorithm (IOR).

Keywords—Path planning, Diffusion, Potential field, Local rules, Parallel architecture.

1. INTRODUCTION

Research in path planning and path finding is encountered in such different disciplines as operations research, robotics, artificial intelligence, psychology, and biology, and is concerned with finding the least-cost path between two states of a system and applied to abstract problems like finding the best way in financial management of a company, as well as to more concrete topographic navigation problems. For a navigating autonomous agent, like a robot in a warehouse or an animal in its everyday surroundings, the knowledge of how to plan trajectories on the basis of information provided by the environment is of great importance for fitness and survival. In this paper we confine our interest to path planning with complete information.

*Author to whom all correspondence should be sent.

This work is supported by BMFT Grant No. 01 IN 104 B/1. We want to express our thanks to A. Baker for proofreading the English manuscript.

Mathematical and engineering theories use mainly analytical top-down approaches to solve this problem, for example, the configuration space method [6], a kind of edge-following algorithm, and the free-space approaches, like generalized cones [7], Voronoi-diagrams [8] or C-cells [9] which belong to the convex decomposition methods. All of these algorithms generate a network of decision points with graph search algorithms working upon it, which possibly presents a bottleneck in the performance of the path planning algorithm as a whole.

An alternative approach makes use of the properties of natural dynamic systems. It comprises algorithms using an idea which may be called the 'dynamic-system metaphor.' The dynamic-system approach is explicitly used, for example, in the reaction-diffusion model of Steels [3,4], in the fluid-dynamics model described in [10], in the retraction process in the neural network developed in [11], and in the Hopfield network with fluid properties used in [12]. The use of potentials for obstacle avoidance, independently developed by Krogh [13] and Khatib [14], can also be classified as a kind of 'dynamic-system approach' since it involves the use of the metaphor of attractive and repulsive forces between electrostatic charges.

This paper describes several path planning algorithms using the dynamic-system metaphor. The simulations show that the generation of path forms similar to those generated by the human probands is possible on the basis of local rules. We focus on the question of how to influence and manipulate gradually the form of the path. We also investigate the decision-making abilities of the algorithms in the case when they choose one out of several path alternatives.

2. BASIC EXPERIMENTAL APPROACH

The experimental paradigm and the results of the psychometric experiments are described in detail in [1,2] and are now sketched briefly. On a computer monitor the probands are confronted with 2-dimensional obstacle situations which have to be avoided while planning and executing a path from a defined start point to a defined goal point. The probands always have to judge two possible path alternatives, taking into account one of the three pre-specified decision criteria, namely, length, duration of movement, and comfort of the path. The difficulty of the imagined paths is characterized by a psychometric decision value, the point of subjective equality, which can be regarded as a turnover point in the decision-making process. In one series of experiments only the 'mental trajectories' are considered, while in another experiment real trajectories are registered on the monitor. Figure 1 shows schematically three examples using the same start and goal points, but three different criteria. As can be seen, the curvature depends on the criterion chosen. On the basis of earlier proposals, we developed several algorithms which can be used to solve the path finding problem. These algorithms will be presented here. They provide tools for later simulations of the results of our psychophysical experiments.

3. MODELING FRAMEWORK

This section outlines the general framework which we developed for path planning based on the dynamic-system metaphor, and which is used for the different computer simulations described in this paper. The dynamic-system approach is a bottom-up and synthetic approach where the path planning problem is mapped into the parameter configuration of a dynamic system. The dynamic properties and the system's inherent capacities of self-organization should finally yield a numerical solution to the problem which, in the case of path planning, is a path connecting initial and final positions.

All computations work on a 2-dimensional map, the cell-matrix M, which consists of $i * j$ cells m_{ij}. This map can be regarded as a specialized world model. Each cell is connected to all its orthogonal and diagonal direct neighbors, represented by the set N_{ij}, forming a chessboard connectivity. The state of each cell m_{ij} at iteration cycle t is characterized by an activation value $a(m_{ij}, t)$ and a marker, which indicates whether the cell is a border cell, an obstacle cell, a start cell m_{start}, a goal cell m_{goal}, or a 'neutral' cell, with no special meaning attached to it. Each

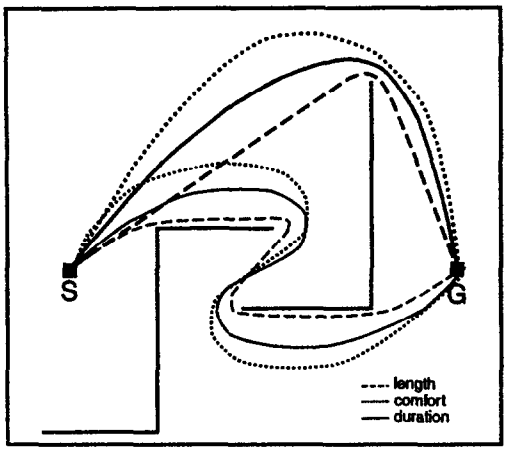

Figure 1. Typical obstacle situation as used in the psychometric experiments. Three examples of paths chosen by a human subject under three different prescribed decision criteria: select a short path (dashed line), a fast path (dotted line), or a comfortable path (continuous line). For all cases both path alternatives are shown.

cell only knows about its own state and the states of its direct neighbors. The grid is updated iteratively until a pre-specified stop criterion is met. Consistent with the theory of cell automata (e.g., [15]), the updating process follows local rules, where the new activation of a cell m_{ij} is essentially a function of the old activations of m_{ij} and its direct neighbors:

$$a(m_{ij}, t+1) = f(a(m_{ij}, t), a(m_{kl}, t)) \tag{1}$$

with m_{kl} belonging to the set of neighboring cells N_{ij}. Using an appropriate function f, this update rule causes a spreading-activation-like behavior which emerges from the system's dynamics.

Obstacles can be implemented by the definition of forbidden areas [6] or by the use of artificial repulsive potentials, mentioned above. We create forbidden areas to mark those cells which are not permitted to lie on a path. The initial activations of these cells are kept constant, but they still belong to the neighborhood N_{ij} of all neighboring cells m_{ij}.

The activation values of the cells can be used for path finding in two different ways, depending on the definition of the update function f. If the resulting activation distribution produced by the algorithm represents a gradient field, a local path finding algorithm can be applied to the grid, following the gradient in the direction of steepest ascent. On the other hand, following the ideas described in [5], an 'optimal area' can be computed. This is a set of cells which contains all cells lying on an optimal path between start and goal. It can be filtered out quickly from the grid by a thresholding process. Further developments of this wave-spreading approach will be introduced in this paper. In principle all algorithms can be computed in parallel. Our calculations are simulated by conventional sequential programming methods. The simulation program is written in 'C' and runs on a PC and a SUN Workstation.

4. DIFFUSION IN POTENTIAL FIELDS (DIP)

Diffusion is a mechanism well known from the natural sciences. In physics and chemistry, it is important for the establishment and maintenance of concentration equilibriums. In biology, the dynamic properties of diffusion in nonequilibrium states are more interesting. The principle of diffusion is used by organisms for information transfer, for example, in the context of finding food or a partner. In ontogeny, diffusion mechanisms are used to explain pattern formation processes (see [16]). Steels [3,4] has shown the applicability of diffusion for path finding problems, not investigating in detail the form of the generated paths. This approach was also proposed in [17,18]

for the purpose of path planning. Issues of mathematical treatment and technical implementation of a diffusion gradient are discussed there in detail. In [17,18] the authors already thought about possibilities of modifying the form of the path. As a solution to the problem, they propose a strategy of modifying the steepest ascent strategy searching for a path in a second step after the generation of the diffusion gradient. In contrast to this approach, we will show in this chapter how to influence explicitly the development of the form of the diffusion gradient, so that the steepest ascent strategy can be kept constant. This method allows us to modulate gradually the form of the generated paths. We also followed the proposal of Steels [3,4]. But because the authors mentioned have not discussed some of the properties of the algorithms in detail, these properties will be illustrated in the form of some examples. This simplifies the discussion of our extended version of the algorithm presented below.

The grid is initialized as follows. The goal point is kept constant and set to a high positive value $d_{\max}$ and all other cells m_{ij} are set to 0. The obstacle and the border cells are specified as the forbidden cells and therefore maintain their zero activation throughout the computation. All other cells change their activations according to the update rule:

$$a(m_{ij}, t+1) = \frac{1}{9} * a(m_{ij}, t) + \sum_{kl}(a(m_{kl}, t)) \tag{2}$$

with m_{kl} belonging to the neighborhood N_{ij} consisting of 8 units. Equation (2) calculates for each cell the arithmetic mean value of all activations in the direct neighborhood of m_{ij}, including its own activation. In this way a diffusion gradient is iteratively built up (see Figure 2a).

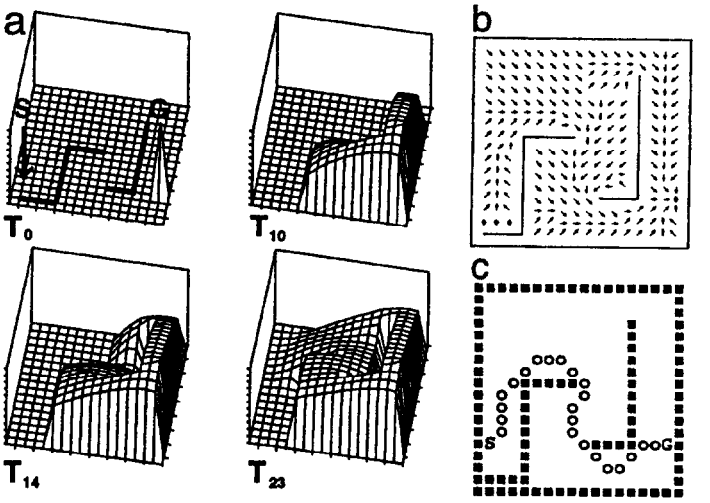

Figure 2. Results of diffusion. (a) Spreading diffusion front depicting the activation distribution at three different moments (clipped logarithmic transformation of the ordinate). T_0: the goal point is initialized with a high positive value, all other cells are set to zero. The start point is marked with an arrow, and the obstacle cells with bold lines. T_{10}, T_{14}, and T_{23}: situation after 10, 14, and 23 iteration cycles, showing the diffusion front shifting around the obstacles. T_{23}: the diffusion front arrives at the start point. (b) Vector diagram calculated from the diffusion gradient at T_{23}, each vector pointing in the direction of the steepest ascent. Points indicate that no neighbor with a larger activation value exists. (c) Generated path when following the arrows shown in (b), cell length = 24.

The iteration continues until either the diffusion front has arrived at the start point m_{start}, using the arrival criterion $a(m_{\text{start}}, t) > 0$, or it stops when the actual iteration number exceeds a pre-specified maximum value so that it is assumed that no path can be found. After the diffusion

process has finished, a path finding procedure using steepest-ascent search is applied to the grid. As an illustration, Figure 2b shows the steepest ascent for each cell of the obstacle situation used in Figure 2a. Following these vectors results in the path which is shown in Figure 2c (filled squares represent obstacles, circles belong to cells lying on the path connecting start S and goal G). If alternative paths exist, all possible paths will be calculated subsequently.

If the diffusion front arrives at the start point, and a path connecting start and goal can be found, then the number of cells constituting that path, the 'cell length,' is equivalent to the number of iteration cycles. Figure 3a demonstrates that the cell length does not necessarily correspond to the geometrical minimum.

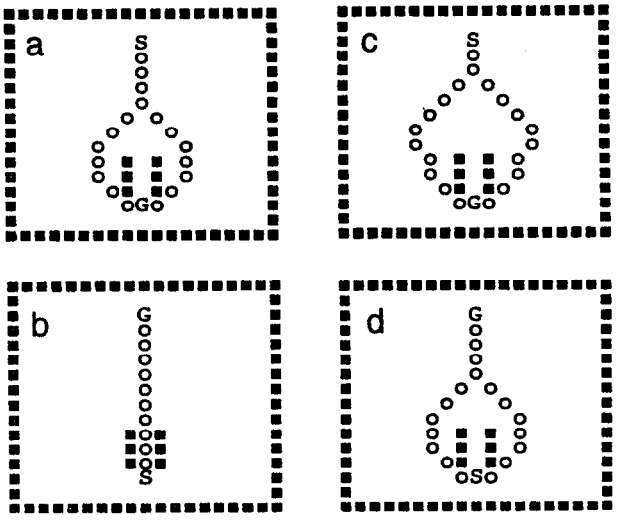

Figure 3. Path generation with the diffusion approach. (a) Using the arrival criterion (12 iteration cycles). (b) Same obstacle situation as in (a), but with start and goal exchanged (11 iteration cycles). (c) Same obstacle situation as in (a), but with the iteration continued for 88 further time steps. (d) same situation as in (b), but with the iteration continued for 89 further time steps.

Because of the zero-valued activations of the obstacle cells, the diffusion front emerging from the goal exhibits an inherent capacity of keeping some distance from obstacles and thereby smoothing the path form. This 'distance-effect' leads to smaller activations of cells close to the obstacles, depending on the number of obstacle cells in the direct neighborhood. The paths of Figures 2c and 3a show this effect, where the cells adjacent to the obstacles are influenced more strongly than those near a corner of the obstacle.

Figure 3a also exemplifies another effect of the diffusion-algorithm. When several path alternatives are possible, the diffusion front splits up in front of the obstacles and runs into different directions. The superimposition of those 'branches' influences the form of the resulting path. If no such superimposition had occurred, the part of the path behind the obstacle would have had a different form, as will be shown below (Figure 5). Another property of the diffusion-algorithm is its asymmetry. This means that, in general, different path forms are obtained when only start and goal points are exchanged; compare Figures 3a and 3b. (Figure 3c shows that the influence of the distance-effect can increase when the iteration is performed beyond the arrival criterion. The form of the path is generally smoother and also the global form of the path may change; compare Figure 3d and Figure 3b.)

The diffusion-algorithm shows a specific decision behavior when there exist several, i.e., usually two, alternative paths. Two computer simulations very similar to those conducted with humans are shown in Figures 4a and b. Using the arrival criterion, one of the two possible paths is chosen

(Figure 4a). Changing the form of the obstacle by one unit, as marked by an open square, leads to a change in the path chosen (Figure 4b).

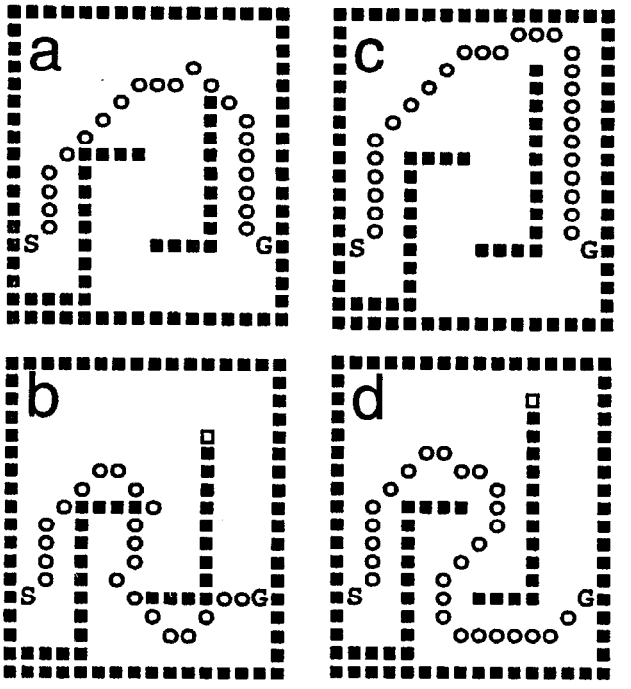

Figure 4. Simulation of the decision-making with the diffusion approach. (a) Using the arrival criterion, one (the upper) of two possible paths is chosen. (b) When the form of the obstacle is changed by one unit (see open square), the lower path is chosen. (c),(d) The same situation as in (a),(b) but using diffusion in potential fields (potential width 1).

The pure diffusion algorithm seems, however, inappropriate for describing some qualitative effects found in the human experiments, such as, for example, more rounded path forms or other effects probably depending on the distance of the path from the obstacles. These results seem to require a more easily adjustable influence than the inherent distance effect. In order to achieve this, we integrated the idea of potential fields [13,14] in the diffusion approach as follows. Before diffusion starts, for each cell m_{ij} a potential value p_{ij} is calculated. The amount of the potential value depends on the number and distance of obstacle cells in the neighborhood which is defined through a potential-function. The use of Gauss-like functions, where the influence of the obstacle cells decreases with increasing distance from these cells, has proved to be reasonable. The potential function is defined up to a certain distance from the obstacle cell which is the origin of the potentials, called the potential width. The potential values p_{ij} are calculated locally for each cell m_{ij} by additively superimposing all potential influences of obstacle cells which lie within a circle around m_{ij} with a radius equal to the potential width. Two methods have been investigated, namely, 'absolute' and 'relative' potentials.

When absolute potentials are used, negative potentials around each obstacle cell are computed, with the absolute maximum value $p_{\max}$ of the potential-function belonging to the obstacle cell itself. The absolute value of $p_{\max}$ should be larger than $d_{\max}$ at the goal, in order to prevent the positive diffusion front from overrunning the negative potentials. Each cell m_{ij} obtains a potential value pa_{ij}. The update rule for the diffusion process changes to

$$a(m_{ij}, t+1) = pa_{ij} + \frac{1}{9} * a(m_{ij}, t) + \sum_{kl}(a(m_{kl}, t)) \tag{3}$$

with m_{kl} belonging to the neighborhood N_{ij}. Subsequent to each iteration cycle, a rectifying procedure is used to cut off the negative activations. The absolute potential metaphorically stands for a sink into which a constant amount of the diffusion front falls and vanishes. The diffusion in the "landscape" produced by the potential fields changes the vector field in a way which qualitatively leads to path forms which generally correspond to those found in the human experiments. However, the influence of the potentials depends on the amount of the arriving diffusion front. Therefore, absolute potentials have the disadvantage that their effects are great near the start, but much smaller near the goal point. As this does not correspond to our experimental results with human probands, an alternative method is investigated which uses relative potentials. They are defined as follows. The potential surfaces are obtained in a similar way to that described above. However, $p_{\max}$ is set to 1 and all potential influences lie in the interval between 0 and 1. The potential values pr_{ij} computed for each cell are obtained by multiplication of all potential influences of the obstacle cells lying within a distance equal to the potential width. Consequently, the potential values of the obstacle and the border cells maintain 1, but values in other cells go to 0 with increasing distance from the obstacles. A factor of $(1 - pr_{ij}) = 0$ indicates that the cell m_{ij} should not belong to the path, while a factor of $(1 - pr_{ij}) = 1$ causes an undisturbed development of the diffusion front. Therefore, at each iteration cycle the diffusion activations are multiplied with the factor $(1 - pr_{ij})$:

$$a\,(m_{ij}, t+1) = (1 - pr_{ij}) * \left[\frac{1}{9} * a\,(m_{ij}, t) + \sum_{kl} (a(m_{kl}, t))\right] \tag{4}$$

with m_{kl} belonging to N_{ij}. When using relative potentials, the number of iteration cycles observed until arrival criterion is the same as without potentials, although the resulting path is longer in the first case. The use of the absolute potential approach generally would have led to a higher number of iterations.

Figures 4c,d show the same obstacle situation as Figures 4a,b, but now relative potentials are used to calculate the gradient. Both paths in Figures 4c,d maintain a larger distance from the obstacles compared to those of Figures 4a,b, thereby producing a longer path. This algorithm also simulates the decision behavior but in general shows a different turnover point.

A disadvantage of the diffusion- and the DIP-algorithm is that superimposition of several diffusion fronts may lead to a gradient field which produces paths whose form appears somewhat unnatural (Figure 3a). After the alternatives have been detected, this could be prevented by letting the diffusion run over the grid a second time, but now blocking all but one path (Figure 5). This would lead to a more 'natural' path form. However, this has the disadvantage that global knowledge is necessary to decide as to how the other paths should be blocked.

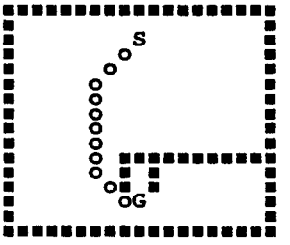

Figure 5. The same obstacle situation as in Figure 3a, but now the right path is blocked by an additional obstacle. This changes the form of the gradient because no superposition occurs.

As an alternative solution, one of the paths as shown in the example of Figure 3a can be used as a rough approximation to the final solution, and the form of this path could be refined

by applying 'Tube-Diffusion.' This means that, based on the original path P, a second diffusion process, using relative potentials and/or iteration beyond the arrival criterion, will be restricted to those cells which lie in the neighborhood of P up to a specified distance d, similar to constructing a tube with P as its central axis. As only a small part of the whole grid has to be updated, this method simplifies the computations. Especially with serial computations and short paths in large grids, this method may considerably save time. A Tube-Diffusion example is shown in Figure 6. Figure 6a shows the result after a normal diffusion. Tube-Diffusion in potential fields is shown in Figures 6b,c with two different forms of the potential function which show the increasing smoothness of the paths.

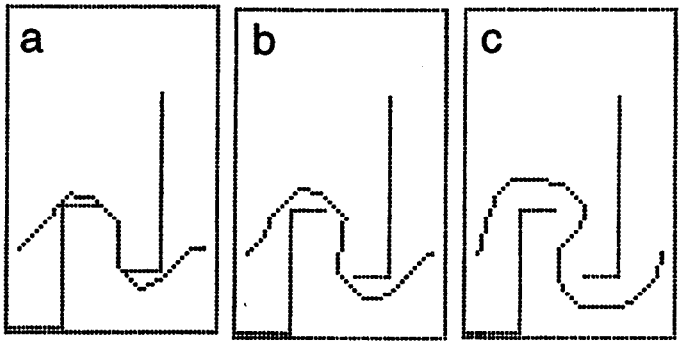

Figure 6. Tube-Diffusion. (a) A path obtained by normal diffusion without potentials using the arrival criterion. (b),(c) Tube-Diffusion in potential fields using the arrival criterion, d = 5. (b) Potential width = 1. Value of the potential function in the direct neighborhood of the obstacles: 0.7. (c) Potential width = 5, values of the potential function with increasing distance from the obstacles: 0.9, 0.6, 0.4, 0.2, 0.1.

5. DIJKSTRA'S ALGORITHM

The algorithm of Dijkstra [19] belongs to the class of blind or exhaustive search algorithms. It provides a solution to the 'single-source shortest-path' problem, working on a directed graph and using an arbitrary cost-function with positive values. Dreyfus [20] has classified it to be the one with the lowest complexity that is guaranteed to find the least-cost path from a start node to every other node.

In [5] Hassoun and Sanghvi have shown how to apply Dijkstra's algorithm to a connectivity matrix instead of a directed graph. They apply the algorithm twice to the grid, automatically generating all path alternatives. Their formulation of the algorithm exhibits its relationship to dynamic programming techniques well known for solving operations research multistage optimization problems [21,22]. A similar algorithm running on a distributed array processor has earlier been described in [23]. Although these variations of the Dijkstra algorithm are tested due to their path finding abilities, no detailed attention has yet been directed to the form of the trajectories. This aspect will be pointed out in the following.

We now briefly introduce the Dijkstra approach used in [5]. Figures 7a–h give a simple example. In addition to the matrix M with the cells m_{ij}, this algorithm uses a second cell matrix C. Their cells c_{ij} have constant positive activations $a(c_{ij})$. C is considered as a cost map such that $a(c_{ij})$ represents the costs belonging to a cell m_{ij}. The sum of the cost values of all cells lying on the path from start to goal should be minimized. Repelling forces around the obstacles can directly be represented in the cost map by high values, which might be called cost-potentials. To initialize matrix M, the goal point is set to zero and kept constant. All other cells are set to a high positive value $u_{\max}$ (see Figure 7b; $u_{\max} = 500$). Matrix C is initialized with the cost values belonging to the cells m_{ij} (Figure 7a). All cells m_{ij} are updated with:

$$a(m_{ij}, t+1) = \min_{kl} \left(a\left(m_{ij}, t\right), a\left(m_{kl}, t\right) + a\left(c_{ij}\right) \right) \tag{5}$$

with m_{kl} belonging to the neighborhood N_{ij}. Thus, the new activation of a cell m_{ij} is the minimum of its own activation and all activations of its direct neighbors summed up with the cost value of cell m_{ij}. According to this rule, the updating is spreading away from the grounded cell. When numerical convergence is achieved, a gradient field results which is represented by the activation matrix M_{goal} at t_{end} (Figure 7c). After convergence is obtained, $a(m_{ij}, t_{\text{end}})$ represents the minimum cost for reaching the grounded cell starting from any cell m_{ij}.

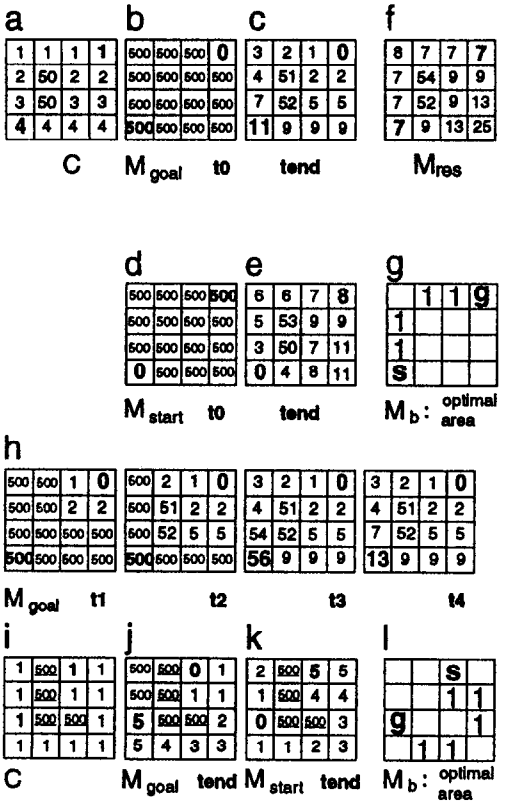

Figure 7. Simple calculation example for Dijkstra's algorithm applied to a 4 × 4 grid. (a)–(h): Arbitrary cost map. (i)–(l): Obstacles are treated as forbidden areas. (a) Cost map C. (b) Initial state of M_{goal}. (c) Final state of M_{goal}. (d) Initial state of M_{start}. (e) Final state of M_{goal}. (f) The sum $M_{\text{goal}} + M_{\text{start}} - C$ results in M_{res} with $u_{\min} = 7$. (g) Using $u_{\min}$ as the threshold the optimal area can be computed. (h) Intermediate states of M_{goal}. (i) Cost map C. (j) Final state of M_{goal}. (k) Final state of M_{start}. (l) Optimal area. Start and goal points are indicated with bold numbers. Here the obstacle cells are assumed to have a value of 50. In the example of Figure 7a the free cells are chosen to have cost values which increase from the upper to the lower margin and thus form a ramp-like cost distribution. Figure 7h shows the development of the activations for the first four time steps. $u_{\max} = 500$, cost values of obstacle cells: 50.

In the second step of the calculation, the start cell instead of the goal cell is set to zero (Figure 7d). Then the above described procedure is repeated until a second gradient surface M_{start} has been built up (Figure 7e). Both activation matrices and the cost matrix are combined, in short: $M_{\text{res}} = M_{\text{start}} + M_{\text{goal}} - C$. The result is shown in Figure 7f. The activation at the start point and the goal point in M_{res} is identical to the minimal activation value $u_{\min}$ in M_{res}. $u_{\min}$ can be used to threshold M_{res} to build up a matrix M_b with the following activations of the cells b_{ij}: $a(b_{ij}) = 1$ if $a(m_{ij}, t_{\text{end}}) = u_{\min}$, $a(b_{ij}) = 0$ otherwise. Figure 7g shows M_b where only the cells b_{ij} with $a(b_{ij}) = 1$. These constitute the so-called optimal areas. The optimal areas include all cells lying on a least-cost path from start to goal or vice versa, comprising all alternative paths

with the same costs. $u_{\min} + a(c_{\text{start}}) + a(c_{\text{goal}})$ is equivalent to the minimum cost of any optimal path connecting start and goal, summing up the costs c_{ij} of all cells lying on the path.

This variant of Dijstra's algorithm described up to this point will be referred to in the following as 'Parallel Implementation of Dijkstra's algorithm' (PID). We want to introduce a modification of this algorithm which shortens the computation time. This modification requires the activation value of the obstacle cells to be held constant at the initial value $u_{\max}$. This will be shown here for the case where all nonobstacle cells obtain the same cost value $a(c_{ij}) = 1$. This means that the paths found will have the shortest cell length. The iteration stops at t_{end} if $a(m_s, t_{\text{end}})$ differs from $u_{\max}$ indicating that the 'wave-front' has arrived at the start (arrival criterion, see Figure 7j). If the activations in the grid are constant over time but the start point has not been reached, then the algorithm stops automatically, indicating that no path can be found since start and/or goal are enclosed by obstacles. Figure 7k shows the result of the second iteration procedure and Figure 7l shows the generated optimal area which indicates that two path alternatives are possible. Since all cells on the path have the same cost value, $u_{\min} + 1$ is equivalent to the number of iteration cycles until the arrival criterion is fulfilled. This procedure can be even more accelerated if only those cells with an activation less than $u_{\max}$ in M_{goal} are taken as the basis for the second iteration step, because only these cells are candidates for the optimal area (compare Figure 7j with Figure 7l). Especially in large grids with a small distance between start and goal, this can shorten the computation time considerably. We will call this accelerated version PIDac.

As was shown above for the DIP-algorithm, PID and PIDac are also able to simulate the decision-making (Figure 8). The obstacle situation is the same as in Figure 4 for the analysis of the diffusion-algorithm.

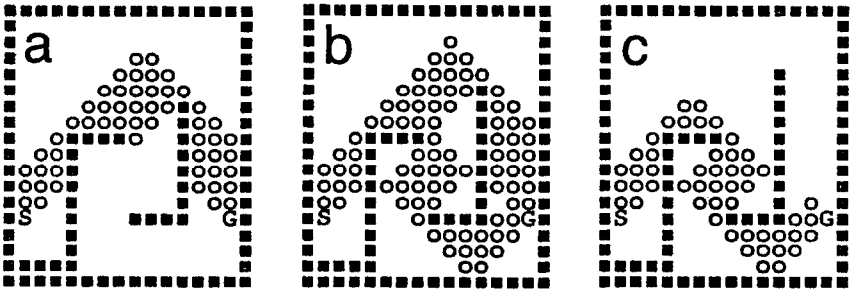

Figure 8. (a) An obstacle situation where PIDac chooses an optimal area containing the upper of two alternatively possible paths. (b) When the obstacle is prolonged by one unit, an optimal area is chosen which contains both alternatives. The length of the right vertical obstacle line in this situation can be regarded as a turnover point in the decision-process of the algorithm. (c) Another prolongation by one unit results in an optimal area which only contains the lower path.

6. INCLUSIVE-OR

In the following, a further development of the PIDac-algorithm, the so-called Inclusive-Or-algorithm (IOR), is described. It is only applicable to maps where the cells are either free or forbidden. This means that the introduction of potentials is not possible. This algorithm is simpler and computationally faster than PID and PIDac.

IOR works on a cell matrix M. Activated cells are indicated by an activation value greater than zero, obstacle cells are prohibited to become activated. Grid M is initialized at t_0 with the goal point set to 1 and all other cells set to zero (Figure 9a). Obstacle cells do not take part in the updating process. They are shown by a dash in Figure 9. The activation of the other cells is

calculated as follows:

$$a(m_{ij}, t+1) = \begin{cases} 1 & \text{if } a(m_{ij}, t) = 0 \text{ and there exists a cell } m_{kl} \\ & \quad \text{belonging to } N_{ij} \text{ with } a(m_{kl}, t) > 0, \\ a(m_{ij}, t) + 1 & \text{if } a(m_{ij}, t) > 0, \\ a(m_{ij}, t) & \text{otherwise.} \end{cases} \tag{6}$$

That means that m_{ij} becomes activated at $t+1$ if m_{ij} is not activated at t but there exists an activated direct neighbor. If m_{ij} was already activated at t, it increases its own activation value by one. If none of these condition is fulfilled, m_{ij} remains unactivated. Consequently more and more cells get caught by the activation front (see Figure 9g). At each moment t, $a(m_{ij}, t)$ shows, in the sense of iteration cycles, how long the cell m_{ij} was already activated. The updating terminates if the activation front has reached the start point so that $a(m_s) > 0$ (arrival criterion). The result is a gradient field M_{goal} (Figure 9b). If the arrival criterion is not fulfilled but no up to now inactive cell becomes activated during the updating of the grid, the algorithm stops automatically since no path can be found.

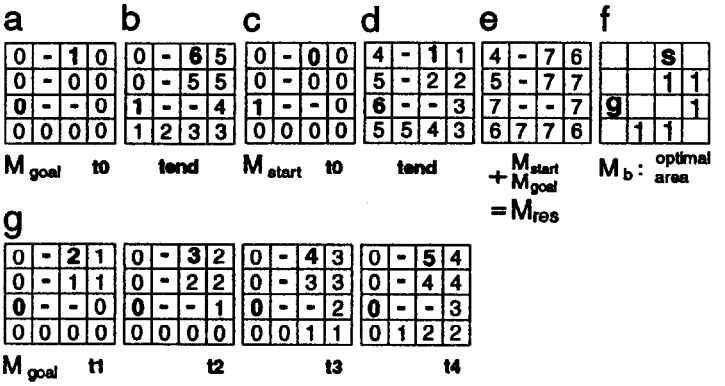

Figure 9. Simple calculation example for the IOR algorithm applied to a 4 × 4 map with 4 obstacles. (a) M_{goal} at t_0. (b) M_{goal} at t_{end}. (c) M_{start} at t_0. (d) M_{start} at t_{end}. (e) Sum of the values shown in (b) and (d). (f) Resulting optimal area which shows two path alternatives with the same minimal cell length. (g) Intermediate states of M_{goal}.

Such wave-spreading results in a gradient to which a local path search algorithm must be applied. This, however, suffers from a computation time problem in search for alternatives. As a solution to these problems, we extend the algorithm by adding a second updating process similarly as it was proposed in [5] for Dijkstra's algorithm.

For this second step the start point is set to 1 and all other cells are grounded (Figure 9c). If we update this matrix, a gradient field M_{start} results (Figure 9d). Both gradient surfaces are combined to a resulting gradient $M_{\text{res}} = M_{\text{start}} + M_{\text{goal}}$ (Figure 9e). Then all cells in M_{res} with the maximum activation are set to one and all other cells are set to zero. Only the ones are shown in Figure 9f. (IOR can be accelerated by restricting the second activation wave, which generates M_{start}, to the area occupied by the activated cells in M_{goal}.)

IOR automatically generates an optimal area which contains all paths of which the cell length corresponds to the maximum activation in M_{res} minus one. The optimal area is identical to that which is obtained from PIDac but can be produced in much less time for the following reasons. Instead of the minimum-operator of PIDac, which has to inspect the activations of all direct neighbors of a cell m_{ij}, the inclusive-or-operator of IOR must only find one activated neighbor. PIDac always continues to apply the minimum-operator. The IOR-algorithm starts slowly, since the inclusive-or-operator has to be applied to all cells. But as more and more cells become activated, the algorithm works faster.

The PID- and the IOR-algorithms do not provide a path but only an optimal area. Therefore, the following question arises. If the optimal area includes more than one optimal path, how can we choose a single one? As an easy solution to this problem one can apply the DIP-algorithm as described above only to the optimal area using the arrival criterion.

7. DISCUSSION AND OUTLOOK

The diffusion-algorithm, which is similar to that proposed in [3,4], provides a gradient field which can be used to find one or several paths connecting start and goal. An advantage of the diffusion-algorithm for a technical utilization is that once a gradient is calculated, it can be used for any starting position. The diffusion approach is therefore applicable to problems such as finding the nearest power source for a mobile agent at a defined position, which is taken as the goal point in the diffusion-algorithm, in the grid. This is a main advantage over the potential approach described in [26] which iteratively builds up a potential surface from start to goal. The diffusion mechanism has the inherent property of keeping a distance from the obstacles. Thereby path forms can be obtained which qualitatively correspond to those produced by human probands. This distance-effect can be increased by applying the diffusion in a potential field (DIP).

By using the algorithms PID/PIDac and the IOR-algorithm instead of a single path, an optimal area is found which consists of all cells comprising all possible optimal paths. The optimality criteria must be specified explicitly in advance. The IOR-algorithm which is computationally much simpler than the algorithms based on Dijkstra's approach can only be applied to the optimality criterion to find the shortest path between start and goal. In order to find the actual path, these algorithms can be combined with the diffusion or the DIP-algorithm which can then be applied to this optimal area. When the geometrical shortest path and not the path with the shortest cell length should be calculated, the PID-algorithm can be used by weighting the costs of diagonal neighbors with the square root of 2. This method has also been used in [23].

Once a path has been found with one of the above described methods, its form can also be modified later on with the Tube-Diffusion algorithm. The algorithms are also successful in finding all possible path alternatives which are equivalent either according to explicitly pre-specified costs (PID and IOR) or according to the implicit optimality criterion inherent in the dynamics of the DIP-algorithm.

In contrast to the diffusion approach, both PID/PIDac- and IOR-algorithms exhibit direction-invariance, i.e., exchanging start and goal does not influence the optimal area. While the diffusion gradient can be influenced by all properties of the obstacle situation, including the size of the grid, the PID/PIDac- and IOR-algorithms produce the same solutions, as long as the optimal area is not effected by the modifications.

The previous chapters have shown that all investigated local algorithms can be used to generate different path forms and therefore might be appropriate to qualitatively simulate the path generation of human probands. The path forms produced by the algorithms described here depend on the obstacle configurations and some pre-specified parameters (e.g., the potential width or the potential function) which can represent the decision criteria in the psychometric experiments described in [1,2]. Especially if all other path alternatives are blocked so that no superimposition of the diffusion fronts deriving from different path alternatives can occur, the forms of the 'naturally' generated paths can be approximated in their main characteristics. Figure 10 shows an example for one test person used in the experiments described in [1,2]. The parameters of the DIP-algorithm in this example are adjusted in such a way that the algorithms produce path forms comparable to those generated by the test person. Although there is qualitative agreement, the paths produced by humans are still smoother.

For a more detailed modeling of the paths forms generated by the test persons, a better adjustment of the relative potentials and the parameters of the DIP algorithm would be helpful.

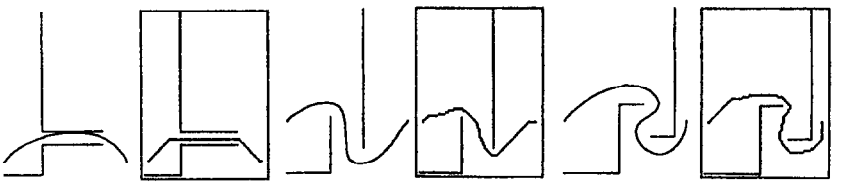

Figure 10. Comparison of path forms generated by a human proband (left) and the diffusion-algorithm (right). The paths are shown for three obstacle situations and the decision criterion comfort. The resulting path is taken as the initial path P for Tube-Diffusion with 500 iteration cycles and the parameter $d = 5$. The potential width is set to 3 and with increasing distance from the obstacle cells the distance values are 0.75, 0.3 and 0.1.

It might also be necessary to include one or the other global parameter. Up to this point our computer simulations already generate adequate solution on a qualitative and somewhat coarse level of description. For example, it might be an interesting approach to investigate a two-step mechanism, where global information is used to build up a temporary path composed of landmarks at a low level of resolution, which is then taken as the input of a second process of relaxation on the basis of local information as they are described in this paper. Such two-step relaxation processes have already been used for the sake of path planning, for instance, in potential field approaches [24,25]. In this way the results of the computer simulations described in this paper provide the basis for the development of algorithms modeling human path planning behavior using mainly local rules working upon a cell-based architecture.

REFERENCES

1. K. Dautenhahn, Pfadplanung beim Menschen, Doctoral Dissertation, appeared as: Materialienband Nr. 63 des Forschungsschwerpunktes Mathematisierung der Einzelwissenschaften der Universität Bielefeld, (1993).
2. K. Dautenhahn, H. Cruse and Th. Kindermann, Parameters for human path planning, *Psychological Research*, (submitted).
3. L. Steels, Steps towards common sense, In *Proc. ECAI,* August 1–5, Munich FRG, pp. 49–54, (1988).
4. L. Steels, Exploiting analogical representations, *Robotics and Autonomous Systems* **6**, 71–88, (1990).
5. A.H. Hassoun and A.J. Sanghvi, Fast computation of optimal paths in two- and higher-dimension maps, *Neural Networks* **3**, 355–363, (1990).
6. T. Lozano-Pérez and M.A. Wesley, An algorithm for planning collision-free paths among polyhedral obstacles, *Communications of the ACM* **22** (10), 550–570, (1979).
7. R.A. Brooks, Solving the find-path problem by good representation of free space, In *Proc. AAAI-82,* August 18–20, Pittsburgh, PA, pp. 381–386, (1982).
8. D.A. Miller, A spatial representation system for mobile robots, In *IEEE International Conference on Robotics and Automation,* March 25–28, St. Louis, pp. 122–127, (1985).
9. R. Chatila, Path planning and environment learning in a mobile robot system, In *Proc. ECAI,* August 1982, Amsterdam, pp. 211–215, (1982).
10. D. Keymeulen and J. Decuyper, A flexible path generator for a mobile robot, In *Proc. Fifth ICAR,* June 19–22, Pisa, Italy, Volume 2, pp. 1069–1073, (1991).
11. W. Shen, J. Shen and J.P. Lallemand, Finding the shortest path by use of neural networks, In *Proc. Fifth ICAR,* June 19–22, Pisa, Italy, Volume 2, pp. 1164–1169, (1991).
12. G. Lei, A neuron model with fluid properties for solving labyrinthian puzzle, *Biological Cybernetics* **64**, 61–67, (1990).
13. B.H. Krogh, A generalized potential field approach to obstacle avoidance control, In *Robotics Research: The Next Five Years and Beyond,* August 14–16, Bethlehem, PA, (1984).
14. O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, In *IEEE International Conference on Robotics and Automation,* March 25–28, St. Louis, pp. 500–505, (1985).
15. E. Rietman, *Exploring the Geometry of Nature: Computer Modeling of Chaos, Fractals, Cellular Automata and Neural Networks*, Windcrest Books, (1989).
16. H. Meinhardt, *Models of Biological Pattern Formation*, Academic Press, (1982).
17. G. Schmidt and K. Azarm, Mobile robot navigation in a dynamic world using an unsteady diffusion equation strategy, In *Proc. of 1992 IEEE International Conf. of Intelligent Robots and Systems,* July 7–10, Raleigh, NC, (1992).

18. G. Schmidt and W. Neubauer, High-speed robot path planning in time-varying environment employing a diffusion equation strategy, In *Robotic Systems*, (Edited by Tzafestas), pp. 207–215, Kluwer Academic Publishers, (1992).
19. E.W. Dijkstra, A note on two problems in connection with graphs, *Numerische Mathematik* **1**, 269–271, (1959).
20. S.E. Dreyfus, An appraisal of some shortest-path algorithms, *Operations Research* **7**, 395–412, (1969).
21. E.M.L. Beale, *Introduction to Optimization*, Wiley and Sons, New York, (1988).
22. R. Bellman, On a routing problem, *Quarterly of Applied Mathematics* **16** (1), 87–90, (1958).
23. C.M. Witkowski, A parallel processor algorithm for robot route planning, In *Proc. Eighth IJCAI*, August 8–12, Karlsruhe, Germany, pp. 827–829, (1983).
24. B.H. Krogh and C.E. Thorpe, Integrated path planning and dynamic steering control for autonomous vehicles, In *International Conference on Robotics and Automation*, Silver Spring, MD, 3 1986, pp. 1664–1669, (1986).
25. Y.K. Hwang and N. Ahuja, A potential field approach to path planning, *IEEE Transactions on Robotics and Automation* **8**, 23–32, (1992).
26. E. Praßler, Robot navigation in unknown terrains—A massively parallel approach, In *7. Fachgespräch über Autonome Mobile Systeme*, Karlsruhe, pp. 187–200, (1991).